Voting-mechanism based ensemble constraint handling technique for real-world single-objective constrained optimization

Xupeng Wen
School of Traffic and Transportation Engineering
Central South University
Changsha, China
wenxupeng@csu.edu.cn

Guohua Wu*
School of Traffic and Transportation Engineering
Central South University
Changsha, China
guohuawu@csu.edu.cn

Mingfeng Fan
School of Traffic and Transportation Engineering
Central South University
Changsha, China
mingfan@csu.cn

Rui Wang
College of Systems Engineering
National University of Defense Technology
Changsha, China
ruiwangnudt@gmail.com

Ponnuthurai Nagaratnam Suganthan
School of Electrical Electronic Engineering
Nanyang Technological University
Singapore
epnsugan@ntu.edu.sg

***Abstract*—Constraint handling techniques are of great significance in efficiently solving constrained optimization problems. This paper proposes a novel ensemble framework for constraint handling techniques based on voting-mechanism, in which four popular constraint handling techniques are included. Each of the constituent constraint handling techniques votes for the solutions at each generation based on its own rules. Solutions getting more votes are regarded as promising individuals and survive to the next generation. This ensemble framework based on voting-mechanism reflects the collective wisdom in decision-making of human beings. In addition, a differential evolution (DE) variant is designed as the search engine, in which four search strategies are combined to generate new individuals and maintain the balance between diversity and convergence of the population. The proposed algorithm has been tested on 57 real world single-objective constraint optimization problems and 7 problems are selected using variable reduction strategy (VRS). The experiment shows that the proposed algorithm achieves competitive performance, indicating that the voting-mechanism ensemble constraint handling technique combine DE algorithm together can effectively deal with constrained optimization problems.**

Keywords—Constraint-handling, Differential Evolution, Voting-mechanism

I. INTRODUCTION

In the real world, there are a lot of complex constrained optimization problems need to be optimized, such as job scheduling, intelligent control, traffic optimization and task allocation[1][2]. For this reason, the IEEE Congress on Evolutionary Computation (CEC) holds the competition for single-objective constrained optimization, aiming to develop more excellent algorithms to solve practical problems. Abhishek et al.[1] collected 57 real-world single-objective constrained optimization problems as the benchmark for the competition and provided the baseline solutions using several state-of-the-art algorithms. Without loss of generality, a constrained single-objective optimization problem (CSOP) includes equality constraints and inequality constraints can be formly described as:

$$\begin{aligned} &Minimize: f(\mathbf{x}) \\ &subject\ to: g_i(\mathbf{x}) \le 0, i = 1,...,p \\ &\qquad h_j(\mathbf{x}) = 0, j = p+1,...,m \\ &\qquad \mathbf{x} = (x_1, x_2, ..., x_n) \end{aligned} \tag{1}$$

where f is the objective function, $(x_1, x_2, ..., x_n)$ are the variables. $g_i(\mathbf{x})$ is the i-th inequality constraint and $h_j(\mathbf{x})$ denotes the j-th equality constraint. The equality constraint $h_j(\mathbf{x})$ is generally transformed into inequality constraint and can be defined:

$$g'_j(\mathbf{x}) = \max(|h_j(\mathbf{x})| - \sigma, 0), \quad j = p+1,...,m \tag{2}$$

where σ is a tolerance parameter and set to 0.0001 according to [1]. The constraint violation of a solution $\mathbf{x}$ is the weighted average of violations over all constraint s:

$$v(\mathbf{x}) = \frac{\sum_{i=1}^{p} w_i (g_i(\mathbf{x})) + \sum_{j=p+1}^{m} w_j (g'_j(\mathbf{x}))}{m} \tag{3}$$

Recently, ensemble optimization algorithms are attracting an increasing attention and show impressive performance in dealing with complex optimization problems [2],[4],[5]. Especially, Mallipeddi, et.al. [14] proposed an ensemble of constraint handling techniques, which is very competitive in solving different kinds of constrained optimization problems. Inspired by the human wisdom in collective decision-making, this paper proposes a novel ensemble framework based on voting-mechanism, which brings together multiple constraint handling techniques to deal with constraints cooperatively. Each constraint handling technique uses its specific rules to handle constraints during the optimization process and select superior solutions via the voting manner like an expert. The

978-1-7281-6929-3/20/$31.00 ©2020 IEEE

solutions getting more votes from the experts (i.e., the constraint handling techniques) will survive to the next generation.

The proposed ensemble constraint handling technique is combined with an ensemble differential evolution (DE) variant to solve constrained optimization problems. In the ensemble DE variant, several search strategies are included. Ensemble of multiple mutation strategies is a promising paradigm in designing versatile DE algorithms [4]. For example, Mallipeddi et al.[23] proposed an adaptive differential evolutionary algorithm EPSDE, in which all mutation strategies and control parameters compete with each other in the evolutionary process to produce offspring. Inspired by EPSDE, Wang et al.[5] proposed the CoDE algorithm which combined three search strategies and three control parameter settings to generate new solutions. Wu et al.[6] proposed an ensemble framework including three indicator subpopulations and a reward subpopulation. Each indicator subpopulation is assigned a mutation strategy, while the larger reward subpopulation is assigned the mutation strategy that currently performs best and would be given more computational resources adaptively during the evolutionary process. Moreover, they proposed the EDEV algorithm in [7], which realizes the ensemble of several efficient DE variants rather than a set of mutation strategies, including JADE[19], CoDE[5] and EPSDE[23], and the efficient algorithms getting more computing resources.

In addition, we introduce the variable reduction strategy (VRS) to reduce the number of equality constraint and variables according to [22]. Through using VRS, the complexity of the problem can be effectively reduced and the search engine can obtain high-quality solution efficiently.

The rest of the paper is structured as follows. Section II briefly introduces four classical constraint handling techniques that are integrated in the proposed ensemble framework. In section III, the voting-mechanism based ensemble constraint handling technique framework is described. In section IV, experimental results are reported and discussed. The final section concludes the paper.

II. RELATE WORK

As mentioned above, a constrained optimization problems (COPs) can be solved by evolutionary algorithms with the assist of constraint handling techniques. In this paper, we propose a voting-mechanism based ensemble framework which form four constraint handling techniques in an ensemble, including Self-Adaptive Penalty (SP), Superiority of Feasible Solutions (SF), Stochastic Ranking (SR) and ε-Constraint (EC). We briefly review these four popular constraint handling techniques in this section.

A. Self-Adaptive Penalty (SP)

The core idea of SP is adding a penalty term $p(\mathbf{x})$ to the objective function $f(\mathbf{x})$ to construct the penalty fitness function $F(\mathbf{x})$. Thus, the constrained optimization problem is transformed into an unconstrained problem. The penalty fitness function is defined as:

$$F(\mathbf{x}) = f(\mathbf{x}) + \alpha \sum_{i=1}^{m} r_i G_i(x), \sum_{i=1}^{m} r_i = 1 \quad (4)$$

where $G_i(x)$ represents the violation of constraint ,r_i is the number of constraint violation levels defined for each constraint, α is the punish coefficient.

B. Tessema et al.[8] proposed an adaptive penalty function method to dynamically adjust the number of penalties for different infeasible individuals. The fitness value is calculated considering constraint violation and the number of penalty. If there are fewer feasible solutions, then the individual with higher constraint violation receives more penalties. If there are more feasible solutions, the individual with high fitness gets less penalties. An adaptive penalty function[9] is proposed to dynamically adjust the coefficients of penalty terms by obtaining useful information in optimization process. The penalty degree of infeasible solution depends on the degree of constraint violation.

B. Superiority of Feasible Solutions (SF)

The constraint handling technique of Superiority of Feasible Solutions proposed by Zielinski [10], which is conformed the following three rules in comparing two solutions. 1). If both solutions are feasible, the one with smaller objective function value is better; 2). If both are infeasible solutions, then the one with less constraint violation is better; 3). If one solution is infeasible solution and another is feasible solution, then the feasible solution is better. However, if the solution space is extremely complex and the feasible region accounts for a small proportion of the search space, it may be not enough to simply category the solutions into the feasible solution and the infeasible solution. Considering this, Cui[11] proposed the concept of relative feasibility to compare and select solutions. If both solutions are not feasible, they are compared first based on the relative feasibility and the one with higher relative feasibility is better. If the relative feasibility of both solutions is equal, then the one with smaller constraint violation is superior.

C. Stochastic Ranking (SR)

Runarsson[12] proposed Stochastic ranking (SR), which introduces a probability *pf* to compare the two solutions based on the objective function value or constraints violation. It makes use of probabilistic parameter *pf* to balance objective function values and constraint violations of solutions. However, this algorithm raises a new problem on how to set the *pf* value appropriately. In the general SR algorithm, the *pf* value is set to 0.475. Further research[13] suggested that it is better that probability *pf* changes adaptively during evolutionary process, which is then combines with the DE algorithm and comes to the DSS-MD algorithm.

D. ε-Constraint (EC)

ε-Constraint[14] is an efficient constraint handling method initially proposed by Takahama. ε-Constraint can be viewed as a general version of SF. ε is a key coefficient adjusting the solution comparison process in ε-Constraint technique. ε is set by a piecewise function as below.

$$\varepsilon(k)=\begin{cases}\varepsilon(0)(1-\dfrac{k}{T_c})^{cp}, 0<k<T_c\\ 0, \quad k\geq T_c\end{cases} \tag{5}$$

To resolve the contradiction between the convergence of and the diversity of population in ε-constraint technique, Takahama[15] proposed a graded ε-constraint technique combined with DE algorithm.

III. ALGORITHM FRAMEWORK

Evidences show that there is no single constraint handling technique that is the most appropriate for all constrained optimization problems. Ensemble of multiple constraint handling techniques is becoming a promising way to get desired performance in solving a variety of constrained optimization problems. For instance, Mallipeddi et al.[16] proposed an ECHT method, in which four popular constraint handling techniques are integrated by partition the population into four subpopulations. Tasgetiren and Suganthan et al.[17] proposed the eDE algorithm, which adopts different search strategies and constraint handling techniques in different optimization processes to generate offspring solutions. In this work, we propose a novel general framework for the ensemble of constraint handling based on voting-mechanism. The voting-mechanism is like the voting behavior human experts and reflects collective wisdom of human beings, which realizes to handle the constraints in a robust manner.

A. The proposed ensemble constraint handling technique based on voting-mechanism

The proposed framework for the ensemble of constraint handling techniques based on the voting-mechanism is named VMCH for short. Each constraint handling technique is considered as an expert and compare and select solution via the voting behavior. Constraint handling experts compare parental and offspring solutions with their specific rules. If a solution is considered being better by a constraint handling technique, it will get a vote from this technique. Based on the voting results, the new solution with more votes is passed to the next generation. In addition, the proposed constraint handling technique is integrated with an efficient DE variant, in which four mutation strategies are included to produce high-quality offspring solutions. The pseudo-code of the main procedure of VMCH is given in Algorithm 1.

Firstly, the parameters of population size and the number of iterations are initialized (line 1). The initial population is generated (line 2). Then, a probabilistic approach is introduced to select a mutation strategy dynamically, in which the probability is updated by *eq*.(10) (line 5). The framework uses historical information to generate parameters F_i and CR_i according [18] (line 6). Offspring individual is generated based on the selected mutation strategy and parameters (line 7). All solutions are voted by four kinds of constraint handling techniques (line 8). The detailed process is clearly described in the *ExpertVote* algorithm. The promising historical parameters are record for later reuse (line 9). The probability q_l and successful count *ns* are updated (lines 10). Population P is updated (line 12). The currently optimal solution is updated if there is a better solution find (line 14).

Algorithm 1：The Main Procedure of VMCH

Input : H: the set of constraint handling techniques (SP, SF, EC and ER)
Output : S_{best}
1 Initialize: population size N, probability q_l for $l=1,2,3,4$
2 generate an initial population P
3 **while**($FES<Max_{FES}$)
4 **for each** p_i in P
5 choose a solution generation strategy according to q_l for $l=1,2,3,4$
6 generate F_i and CR_i based on S_F and S_C
7 o_i←generate a new individual with selected stratgy and parameters
8 execute *ExpertVote* (H, p_i, o_i, P') to select solution and update to P'
9 record promising F_i and CR_i to S_F and S_C, respectively
10 update *ns* and q_l according *eq*.(10)
11 **end for**
12 update P← P'
13 **end while**
14 S_{best}←choose the best solution in P

Inspired by human voting behavior, a voting-mechanism based ensemble framework is proposed to get an ensemble constraint handling technique including multiple constraint handling techniques. Each constituent constraint handling technique is exactly an expert. The framework is very flexible and it is theoretically able to include any number of constraint handling technique. Each constraint handling technique compares the parental and offspring solutions according to their special rules, i.e., voting either parent solution or the offspring solution to survive to the next generation. The voting result of the i-th expert v_i on one pair solution is 1 or 0, in which value 1 represents that the offspring individual is better and value 0 means that the parental individual is preferred. For example, $v_i^k=1$ means that the i-th expert vote the offspring individual on k-th pair. Algorithm 2 gives the pseudocode for voting procedure of the constraint handling techniques.

In Algorithm 2, we describe the algorithm of *ExpertVote*, in which four experts of constraint handling techniques work cooperatively, that is, Self-Adaptive Penalty (SP), Superiority of Feasible Solutions (SF), ε-Constraint (EC) and Stochastic Ranking (SR). First, initialize the vote values to zero for both parental and offspring solution (line 1). Each constraint handling technique is employed to vote between the parental solution and the offspring solution and vote results are computed accordingly (lines 3-7). The individuals with more votes are selected to enter the next generation and the population P' is updated with the superior solution being inserted (lines 9-13).

Algorithm 2：*ExpertVote* algorithm

Input：*p:* a parental solution
o: an offspring solution
H: a set of constraint handling techniques
P': the population to be updated
Output: Updated population *P'*
1 Initialize V_o and V_p to zero
2 **for** each *h* in *H*
3 **if** *o* is better than or equal to *p* according to the rules of constrain handling technique *h*
4 $V_o = V_o + 1$
5 **else**
6 $V_p = V_p + 1$
7 **end if**
8 **end for**
9 **if** $V_o \geq V_p$
10 insert *o* into population *P'*
11 **else**
12 insert *p* into population *P'*
13 **end if**

B. Variable Reduction Strategy

In order to decrease the complexity of problems, we introduce Variable Reduction Strategy (VRS) by using part of variables to represent the other part of variables. Thus, the variables are reduced and the decision space of this problem is decreased, what's more, the search efficiency of the algorithm VMCH is improved. In particular, VRS is defined as follow：

Assume A is collection of variables, $A=\{x_k \mid k=1,2,\cdots\}$, A_j denodes variable collection of *j-th* equality variable, $A_j \in A$. For the equality $h_j(\mathbf{x})$, if x_k can represent by:

$$x_k = R_{k,j}(\{x_l \mid l \in A_j, l \neq k\}) \tag{6}$$

where x_k is reduced variable, x_l is core variable. x_k can be calculated through the relation $R_{k,j}$. Thus, the variable x_k is reduced and the equation $h_j(\mathbf{x})$ is also reduced. Meanwhile, a boundary constraint associated with x is added:

$$\min x_k \leq x_k = R_{k,j}(\{x_l \mid l \in A_j, l \neq k\}) \leq \max x_k \tag{7}$$

In the process of solving the nonlinear equation system, the constraint conditions of eq.(7) can be handled without affecting the complexity of the algorithm.

C. DE variant with multiple mutation strategies

We adopt four solution search strategies as search engine in this study. The search strategies are dynamically selected and the related parameters are adapted during the optimization process. The four efficient solution search strategies are:

(a) *DE/current-to-pbest/1* and Bin, and (b) *DE/current-to-pbest/1* and Exp. These two variants are derived from *DE/current-to-pbest/1*, which is proposed in [19]:

$$v_{i,G} = x_{i,G} + F_i(x_{pbest,G} - x_{i,G}) + F_i(x_{r1,G} - x_{r2,G}) \tag{8}$$

(c) *DE/randr1*/1* and Bin, and (d) *DE/randr1*/1* and Exp. The mutation strategy of *DE/randr1*/1* is improved from *DE/randr1/1*, which is presented in [20]:

$$v_{i,G} = x_{r1^*,G} + F_i(x_{r2^*,G} - x_{r3^*,G}) \tag{9}$$

The four efficient solution search strategies above is selected via a competition mechanism in Algorithm 1 (line 8), which is based on a probability-based selection strategy proposed in [21]. The search strategy is selected in terms of the probability values at different evolutionary stages and the probability values are dynamically updated. The values of initial probabilities of all search strategies are set to be equal. The probability value is determined based on the performance of each search strategy in the evolutionary process. The update formula of the strategy selection probability is as below.

$$q_l = \frac{n_l + n_0}{\sum_{k=1}^{K}(n_k) + n_0} \tag{10}$$

where, n_l is the number of successful individuals produced by the *l*-th search strategy. To prevent a certain strategy from losing selection opportunity in the evolutionary process, reset q_l to 1/K and n_l to 0 respectively, when value of q_l is less than a threshold value κ.

IV. EXPERIMENTAL STUDIES

A. Experimental setting

To investigate the effectiveness of the proposed algorithm VMCH, we use the competition benchmark in CEC2020 including 57 real-world single-objective constrained optimization problems. The proposed algorithm VMCH integrates four classical constraint handling techniques, including Self-Adaptive penalty (SP), Superiority of Feasible Solutions (SF), Stochastic Ranking (SR) and ε-Constraint (EC). The main parameters of these constraint handling techniques are as follows: Tc = 0.2* Max_{Gen}, cp = 5, Sr = 0.475. In addition, we set the population size to 150. The allowed maximum function evaluations are given in [1]. Besides, The mutation strategy parameter p of *DE/current-to-pbest/ 1* is set to 0.05. In addition, we set assigned K=4 and n_0=2.

B. Variable Reduction Strategy Experiment

In this experiment part, in order to test the performance of variable reduction strategy, we choose RC01, RC02, RC04, RC05, RC6, RC07 and RC09 from CEC2020 competition benchmark to pre-process problems using VRS. Table I shows the reduction of the number of equations and variables by using the reduction strategy. $\boldsymbol{N_v}$ is the original number of variables, $\boldsymbol{N_{v\ (\mathrm{VRS})}}$ is the number of variables after using VRS, $\boldsymbol{N_e}$ is the original number of equality constraints. $\boldsymbol{N_{e(\mathrm{VRS})}}$ is the number of equality constraints after using VRS. Table II shows the results of solving these problems with and without using VRS.

TABLE I. The reduction of equation and variable

Problem	$\boldsymbol{N_v}$	$\boldsymbol{N_{v\ (\mathrm{VRS})}}$	$\boldsymbol{N_e}$	$\boldsymbol{N_{e(\mathrm{VRS})}}$
RC01	9	5	8	3
RC02	11	6	9	3
RC04	6	3	4	1
RC05	9	6	4	1
RC06	38	12	32	8
RC07	48	27	38	16
RC09	3	2	1	0

As seen in Table II, through the using of variable reduction strategy, the proposed VMCH with VRS find better solutions on 5 problems of these 7 problems than without using variable reduction strategy, and problem 02, problem 05, problem 06 and problem 07 are not find optimal solutions. In addition, the result of using VRS has less variance than without using VRS, indicating that the VRS strategy can make the search for solutions more stable.

C. VMCH Experiment and Discussion

For the sake of fair competition, the proposed algorithm runs 25 times on each problem according to the setting of Test-suit in [3]. This algorithm calculates the Best, Worst, Median, Mean and Std of the objective function values f and the constraint violation values v of 25 independent runs, respectively. FR represents the proportion of the runs that at least find one feasible solution. v is the averaged constraint violation of all 25 runs. c is a sequence of three numbers represents the number of times the constraint has been violated on the median solution. The three number indicates that the number of constraints is more than 1.0, between [0.01, 1.0] and less than 0.01, respectively. The experimental results of VMCH algorithm are shown in Table III~Table VIII.

From Table III~Table VIII, the results are shown in bold if the mean value of solutions is close to the optimal solution known up to now. The performance of the proposed VMCH is similar to the algorithms of top3 in the competition of CEC2018, i.e., IUDE, ϵMagES and iLSHADE$_\epsilon$. Due to the complementary between constraint handling techniques, such as feasibility rule and penalty function, have been verified by theoretical. The reason of VMCH is competitive is that these four constraint handling techniques are complement each other in some sense and this mechanism adopts the principle that the minority is subordinate to the majority to vote for the better solution, which has strong robustness for the problem with poor effect of single constraint handling technique. From the Mechanical Engineering problems, the proposed framework achieves competitive result, which the number of variables is less than or 30. However, the proposed VMCH finds few optimal solutions known up to now on these problems in Power System and Livestock Feed Ration Optimization filed, which have more than 50 dimensions. In the view of this phenomenon, the proposed framework is suit for solving low dimensional problems.

The algorithm complexity is measured by the criterion in [1], T_1 is the average computing time of running 100,000 times on the test set, and T_2 is the complete computing time for the algorithm of running 100,000 times on the test set. The complexity of VMCH in this paper is compared with the test algorithm in [1], including IUDE, ϵMagES and iLSHADE$_\epsilon$. The comparison results are shown in Table IX.

TABLE IX. Algorithm Complexity

Algorithm	T_1(sec)	T_2(sec)	$(T_2- T_1)/ T_1$
VMCH	8.57	11.34	0.32
IUDE	8.57	9.93	0.16
ϵMAgES	8.57	12.67	0.48
iLSHADE$_\epsilon$	8.57	15.76	0.84

As shown in Table IX, As a result, compare with the three algorithms, VMCH is particularly efficient in dealing with constrained optimization problems of relatively smaller size. We conclude two reasons as follow: 1). these constraint handling techniques are sensitive and can't handle effective on high dimension problems. 2). these constraint handling techniques underutilize domain knowledge to guide the search algorithm to better solutions, especially in complex, narrow feasible regions with high dimension.

TABLE II. Experimental results of VMCH with VRS or without VRS

		RC01	RC02	RC04	RC05	RC06	RC07	RC09
VMCH with VRS	Best	9.37E+01	6.85E+03	-3.88E-01	-4.00E+02	9.98E-01	1.86E+00	2.56E+00
	Median	2.08E+00	3.07E+03	-3.86E-01	-4.00E+02	1.14E+00	1.52E+00	2.56E+00
	Mean	7.41E+00	3.08E+03	-3.87E-01	-4.00E+02	1.09E+00	1.39E+00	2.56E+00
	Worst	6.68E-01	4.78E+02	-3.85E-01	-4.00E+02	1.76E+00	1.13E+00	2.56E+00
	Std	2.04E+01	1.57E+03	1.13E-02	7.42E-12	5.21E-01	1.34E+00	9.06E-16
	v	0.00E+00	0.00E+00	0.00E+00	0.00E+00	2.96E-02	6.35E-01	0.00E+00
VMCH without VRS	Best	3.55E+02	9.65E+03	-1.67E-01	-4.00E+02	1.76E+00	1.93E+00	2.56E+00
	Median	9.08E+01	8.85E+03	-3.51E-01	-4.09E+02	1.89E+00	1.45E+00	2.81E+00
	Mean	1.12E+02	8.89E+03	-3.39E-01	-4.12E+02	1.83E+00	1.47E+00	2.87E+00
	Worst	0.00E+00	8.21E+03	-3.95E+00	-4.27E+02	2.03E+00	1.28E+00	3.11E+00
	Std	1.19E+02	3.78E+02	4.22E-02	5.29E-01	1.89E+00	1.21E+00	8.58E-01
	v	1.06E+02	1.73E+01	0.00E+00	0.00E+00	5.08E-01	7.96E-01	1.79E-05

TABLE III. EXPERIMENTAL RESULTS OF VMCH ON CEC2020 CSOPS (FROM RC01 TO RC10)

		RC01	RC02	RC03	RC04	RC05	RC06	RC07	RC08	RC09	RC10
Best	*f*	9.37E+01	6.85E+03	-1.88E+04	-3.88E-01	-4.00E+02	9.98E-01	1.86E+00	2.00E+00	2.56E+00	1.08E+00
	v	0.00E+00	2.73E+01	0.00E+00	0.00E+00	0.00E+00	6.17E-02	4.81E-01	0.00E+00	0.00E+00	0.00E+00
Median	*f*	2.08E+00	3.07E+03	-2.07E+04	-3.86E-01	-4.00E+02	1.14E+00	1.52E+00	2.00E+00	2.56E+00	1.08E+00
	v	0.00E+00	1.73E+01	0.00E+00	0.00E+00	0.00E+00	2.65E-02	8.03E-01	0.00E+00	0.00E+00	0.00E+00
Mean	*f*	**7.41E+00**	**3.08E+03**	-2.06E+04	**-3.87E-01**	**-4.00E+02**	**1.09E+00**	**1.39E+00**	**2.00E+00**	**2.56E+00**	**1.08E+00**
	v	0.00E+00	1.65E+01	0.00E+00	0.00E+00	0.00E+00	2.96E-02	6.35E-01	0.00E+00	0.00E+00	0.00E+00
Worst	*f*	6.68E-01	4.78E+02	-2.16E+04	-3.85E-01	-4.00E+02	1.76E+00	1.13E+00	2.00E+00	2.56E+00	1.08E+00
	v	0.00E+00	6.14E+00	0.00E+00	0.00E+00	0.00E+00	1.57E-02	4.84E-01	0.00E+00	0.00E+00	0.00E+00
Std	*f*	2.04E+01	1.57E+03	7.11E+02	1.13E-02	-4.00E+02	5.21E-01	1.34E+00	0.00E+00	9.06E-16	4.53E-16
	v	0.00E+00	5.72E+00	0.00E+00	0.00E+00	0.00E+00	6.71E-03	1.30E-01	0.00E+00	0.00E+00	0.00E+00
FR		100	92	100	100	100	0	0	100	100	100
c		0,0,0	0,0,0	2,0,0	0,0,0	0,0,0	1,0,0	0,0,0	0,0,0	0,0,0	0,0,0

TABLE IV. EXPERIMENTAL RESULTS OF VMCH ON CEC2020 CSOPS (FROM RC11 TO RC20)

		RC11	RC12	RC13	RC14	RC15	RC16	RC17	RC18	RC19	RC20
Best	*f*	1.54E+02	4.00E+00	2.69E+04	6.25E+04	3.00E+03	6.35E-02	1.27E-02	6.37E+03	1.67E+00	2.64E+02
	v	3.56E-01	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
Median	*f*	5.71E+01	2.92E+00	2.69E+04	5.86E+04	2.99E+03	3.24E-02	1.27E-02	6.06E+03	1.67E+00	2.64E+02
	v	2.11E-03	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
Mean	*f*	1.25E+02	**2.92E+00**	**2.69E+04**	5.99E+04	**2.99E+03**	**3.35E-02**	**1.27E-02**	6.06E+03	**1.67E+00**	**2.64E+02**
	v	1.42E-01	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
Worst	*f*	1.21E+02	3.07E+00	2.69E+04	6.02E+04	3.00E+03	3.69E-02	1.27E-02	6.07E+03	1.67E+00	2.64E+02
	v	1.59E-01	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
Std	*f*	2.34E+01	3.54E-01	1.11E-11	1.29E+03	1.04E+00	7.33E-03	1.58E-08	6.23E+01	4.68E-06	1.16E-14
	v	1.14E-01	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
FR		100	100	100	64	100	100	100	100	100	100
c		0,0,0	0,0,0	0,0,0	1,0,0	0,0,0	0,0,0	0,0,0	0,0,0	0,0,0	0,0,0

TABLE V EXPERIMENTAL RESULTS OF VMCH ON CEC2020 CSOPS (FROM RC21 TO RC30)

		RC21	RC22	RC23	RC24	RC25	RC26	RC27	RC28	RC29	RC30
Best	*f*	2.35E-01	5.37E-01	1.76E+01	6.25E+00	1.81E+03	1.63E+02	5.42E+02	1.46E+04	2.96E+06	2.91E+00
	v	0.00E+00	0.00E+00	1.65E-02	0.00E+00	0.00E+00	1.31E-03	0.00E+00	0.00E+00	0.00E+00	0.00E+00
Median	*f*	2.35E-01	5.26E-01	7.90E+00	3.26E+00	1.62E+03	3.95E+01	5.24E+02	1.46E+04	2.96E+06	2.66E+00
	v	0.00E+00	0.00E+00	2.78E-06	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
Mean	*f*	**2.35E-01**	**5.26E-01**	8.82E+00	3.93E+00	**1.62E+03**	6.43E+01	**5.31E+02**	**1.46E+04**	**2.96E+06**	**2.66E+00**
	v	0.00E+00	0.00E+00	8.67E-03	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
Worst	*f*	2.35E-01	5.27E-01	1.04E+01	4.07E+00	1.64E+03	7.05E+01	5.31E+02	1.46E+04	2.96E+06	2.67E+00
	v	0.00E+00	0.00E+00	8.31E-03	0.00E+00	0.00E+00	5.24E-05	0.00E+00	0.00E+00	0.00E+00	0.00E+00
Std	*f*	1.13E-16	2.41E-03	2.61E+00	6.48E-01	4.71E+01	2.97E+01	4.49E+00	6.58E+00	1.43E-09	5.02E-02
	v	0.00E+00	0.00E+00	4.71E-03	0.00E+00	0.00E+00	2.62E-04	0.00E+00	0.00E+00	0.00E+00	0.00E+00
FR		100	100	100	0	100	100	72	100	100	100
c		0,0,0	0,0,0	0,0,0	0,2,0	0,0,0	0,0,0	0,0,0	0,0,0	0,0,0	0,0,0

TABLE VI. EXPERIMENTAL RESULTS OF VMCH ON CEC2020 CSOPS (FROM RC31 TO RC40)

		RC31	RC32	RC33	RC34	RC35	RC36	RC37	RC38	RC39	RC40
Best	*f*	2.57E-15	-3.07E+04	2.95E+00	6.85E+00	-1.99E+00	1.57E+02	2.55E-01	7.90E+00	1.83E+01	7.64E+05
	v	0.00E+00	0.00E+00	0.00E+00	1.12E-01	1.87E-01	1.69E-01	9.86E-02	1.06E-01	1.27E-01	4.20E+01
Median	*f*	0.00E+00	-3.07E+04	2.64E+00	4.11E-02	-7.11E+02	-3.40E+02	-1.26E+01	-1.71E+01	-2.77E+01	3.78E+04
	v	0.00E+00	0.00E+00	0.00E+00	1.07E-04	4.07E-04	7.38E-06	9.95E-06	6.23E-05	1.11E-03	2.15E+00
Mean	*f*	**1.17E-19**	**-3.07E+04**	**2.64E+00**	1.75E+00	-7.02E+02	-2.73E+02	-7.93E+00	-5.73E+00	-1.24E+01	1.76E+05
	v	0.00E+00	0.00E+00	0.00E+00	2.46E-02	2.93E-02	3.49E-02	2.92E-02	4.04E-02	2.92E-02	2.02E+01
Worst	*f*	1.61E-16	-3.07E+04	2.64E+00	2.11E+00	-5.87E+02	-2.42E+02	-7.70E+00	-5.72E+00	-1.29E+01	2.28E+05
	v	0.00E+00	0.00E+00	0.00E+00	3.40E-02	4.27E-02	4.51E-02	2.81E-02	3.88E-02	3.81E-02	1.99E+01
Std	*f*	5.30E-16	5.62E-01	6.24E-04	1.79E+00	2.10E+02	1.31E+02	3.70E+00	6.94E+00	9.48E+00	1.77E+05
	v	0.00E+00	0.00E+00	0.00E+00	3.07E-02	4.35E-02	4.81E-02	2.44E-02	2.61E-02	3.21E-02	1.04E+01
FR		100	100	100	0	0	0	0	0	0	0
c		0,0,0	0,0,0	0,0,0	1,0,0	2,0,0	1,0,0	1,0,0	2,0,0	1,0,0	2,0,0

TABLE VII. EXPERIMENTAL RESULTS OF VMCH ON CEC2020 CSOPS (FROM RC41 TO RC50)

		RC41	RC42	RC43	RC44	RC45	RC46	RC47	RC48	RC49	RC50
Best	*f*	2.69E+05	-6.62E+02	-1.07E+02	-5.45E+03	2.03E+00	1.58E+00	7.93E-01	7.47E-01	7.75E-01	4.28E-01
	v	2.75E+01	3.11E+01	3.13E+01	0.00E+00	1.14E-04	4.17E-05	7.88E-05	1.25E-05	3.66E-05	3.82E-05
Median	*f*	6.43E+04	-2.18E+03	-1.04E+03	-6.05E+03	2.54E-01	1.23E-01	7.30E-02	1.23E-01	7.10E-02	4.70E-02
	v	1.99E+00	5.17E+00	8.25E+00	0.00E+00	6.63E-08	7.44E-08	5.60E-11	5.77E-08	9.95E-08	1.60E-07
Mean	*f*	1.13E+05	-1.42E+03	-5.69E+02	-5.73E+03	7.36E-01	4.74E-01	3.02E-01	3.04E-01	2.38E-01	1.78E-01
	v	2.16E+01	1.73E+01	1.85E+01	0.00E+00	3.91E-06	4.39E-06	2.73E-06	2.34E-06	1.42E-06	2.03E-06
Worst	*f*	1.29E+05	-1.37E+03	-5.74E+02	-5.76E+03	8.51E-01	5.18E-01	3.43E-01	3.32E-01	2.73E-01	1.97E-01
	v	2.02E+01	1.74E+01	1.90E+01	0.00E+00	1.03E-05	7.48E-06	8.21E-06	3.02E-06	5.15E-06	6.12E-06
Std	*f*	5.87E+04	3.52E+02	1.90E+02	1.60E+02	4.84E-01	3.19E-01	1.94E-01	1.66E-01	1.75E-01	1.09E-01
	v	5.97E+00	6.59E+00	5.93E+00	0.00E+00	2.32E-05	1.01E-05	1.78E-05	2.91E-06	8.30E-06	8.86E-06
FR		0	0	0	0	100	0	0	0	0	0
c		1,0,0	2,0,0	1,0,0	1,0,0	0,0,0	0,2,0	1,0,0	1,0,0	0,1,0	0,1,0

TABLE VIII . EXPERIMENTAL RESULTS OF VMCH ON CEC2020 CSOPS (FROM RC51 TO RC57)

		RC51	RC52	RC53	RC54	RC55	RC56	RC57
Best	*f*	2.56E+04	4.54E+04	4.48E+04	2.17E+04	4.47E+04	3.63E+04	4.64E+04
	v	2.47E-06	3.87E-07	9.47E-07	7.64E-05	2.25E+00	1.38E+00	1.64E+00
Median	*f*	7.59E+03	6.42E+03	6.78E+03	4.59E+03	7.26E+03	9.33E+03	5.65E+03
	v	6.77E-10	1.05E-09	2.29E-13	3.27E-11	1.88E-01	4.66E-02	7.47E-02
Mean	*f*	1.27E+04	1.49E+04	2.10E+04	1.37E+04	2.07E+04	1.59E+04	2.09E+04
	v	1.77E-07	3.69E-08	3.30E-08	4.94E-08	5.60E-01	5.00E-01	7.09E-01
Worst	*f*	1.37E+04	1.68E+04	2.36E+04	1.33E+04	1.99E+04	1.85E+04	2.07E+04
	v	3.24E-07	8.48E-08	8.79E-08	3.83E-06	8.10E-01	5.66E-01	8.17E-01
Std	*f*	5.10E+03	9.88E+03	1.26E+04	5.77E+03	9.38E+03	7.67E+03	1.02E+04
	v	5.35E-07	1.11E-07	1.87E-07	1.53E-05	5.57E-01	3.41E-01	4.61E-01
FR		0	0	0	0	0	0	0
c		1,0,0	1,0,0	2,0,0	2,0,0	1,0,0	2,0,0	2,0,0

V. CONCLUSION

In this paper, we propose a novel ensemble framework based on voting-mechanism, which integrates four classical constraint handling techniques, including Self-Adaptive Penalty (SP), Superiority of Feasible Solutions (SF), Stochastic Ranking (SR) and ε-Constraint (EC). These four techniques are viewed as experts and each expert votes for each pair of parent individual and offspring individual. The individual have more votes will be selected to enter to the next generation. The proposed ensemble constraint handling technique based on voting-mechanism overcomes the limitation of single constraint handling technique by integrating the advantages of multiple constraint handling techniques. The algorithm is tested on the test-suite about constrained single objective optimization problems from the real world. The experimental results show that the proposed VMCH has competitive performance in solving constrained optimization problems. In the future, we will develop efficient constraint handling technique to search more efficient solutions.

ACKNOWLEDGMENT

This work was supported in part by the Natural Science Fund for Distinguished Young Scholars of Hunan Province under Grant 2019JJ20026, and Fundamental Research Funds of Central South University.

REFERENCES

[1] CAC Coello. Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: A survey of the state of the art. Computation Methods in Applied Mechanics and Engineering, 2002,191(11-12):1245-1287.

[2] H. Chen, X. Zhu, G. Liu and W. Pedrycz, "Uncertainty-Aware Online Scheduling for Real-Time Workflows in Cloud Service Environment," in IEEE Transactions on Services Computing.

[3] A. Kumar, G. Wu, M.Z. Ali, et.al.A test-suite of non-convex constrained optimization problems from the real-world and some baseline results. Swarm and Evolutionary Computation, DOI: 10.1016/j.swevo.2020. 100693, 2020..

[4] G.Wu, R.Mallipeddi, P.N.Suganthan.Ensemble Strategies for Population based Optimization Algorithms-A Survey. Swarm and Evolutionary Computation, 2018, 2019, 44: 695-711.

[5] Y. Wang, Z. Cai, Q. Zhang, "Differential evolution with composite trial vector generation strategies and control parameters," IEEE Transactions on Evolutionary Computation, vol. 15, no. 1, pp. 55–66, 2011

[6] G. Wu, R. Mallipeddi, P. N. Suganthan, R. Wang, H. Chen, Differential Evolution with Multi-Population Based Ensemble of Mutation Strategies. Information Sciences, 329 (2016): 329-345.

[7] G. Wu, X. Shen, H. Li, H. Chen, A. Lin, P.N. Suganthan, Ensemble of differential evolution variants, Information. Sciences. 423 (2018) 172–186.

[8] B. Tessema, G. G. Yen, "A self-adaptive penalty function based algorithm for constrained optimization," in Proc. IEEE Congress on Evolutionary Computation., Vancouver, BC, Canada, 2006. 246–253.

[9] K. Basheed. An adaptive penalty approach for constrained genetic algorithm optimization. In: Koza JR, et al., eds. Proc. of the 3rd Annual Conf. on Genetic Programming (GP'98)/Symp. on Genetic Algorithms (SGA'98). San Francisco: Morgan Kaufmann Publishers, 1998. 584–590.

[10] R. Zielinski, R. Laur. Constrained single-objective optimization using differential evolution. In: Proc. of the 2006 IEEE Int'l Conference on Evolutionary Computation. Vancouver: IEEE Press, 2006. 223–230.

[11] CG. Cui, YJ. Li, TJ. Wu. A relative feasibility degree based approach for constrained optimization problems. Frontiers of Information Technology and Electronic Engineering, 2010,11(4):249–260.

[12] TP. Runarsson, X. Yao. Stochastic ranking for constrained evolutionary optimization. IEEE Trans. on Evolutionary Computation,2000, 4(3): 284–294.

[13] M. Zhang, WJ. Luo, X. Wang. Differential evolution with dynamic stochastic selection for constrained optimization. Information Sciences, 2008,178(15):3043–3074.

[14] T. Takahama, S. Sakai. Constrained optimization by the constrained differential evolution with gradient-based mutation and feasible elites. In: Proc. of the 2006 IEEE Int'l Conf. on Evolutionary Computation. Vancouver: IEEE Press, 2006. 372–378.

[15] T. Takahama, S. Sakai. Efficient constrained optimization by the ε constrained rank-based differential evolution. In: Proc. of the 2012 IEEE Congress on Evolutionary Computation. Brisbane: IEEE Press, 2012. 1–8.

[16] R. Mallipeddi, P. N. Suganthan. Ensemble of constraint handling techniques. IEEE Transactions on Evolutionary Computation, 2010,14 (4): 561–579.

[17] M. Tasgetiren, P. N. Suganthan, Q. Pan, R. Mallipeddi, S. Sarman. An ensemble of differential evolution algorithms for constrained function optimization. In: Proc. of the IEEE Congress on Evolutionary Computation Vol.5. Barcelona: IEEE Service Center, 2010.967–975.

[18] R. Tanabe, A. Fukunaga, "Success-history based parameter adaptation for differential evolution," in Evolutionary Computation (CEC),2013 IEEE Congress on. IEEE, 2013, pp. 71–78.

[19] J. Zhang, A. C. Sanderson, "Jade: adaptive differential evolution with optional external archive," IEEE Transactions on evolutionary computation, vol. 13, no. 5, 2009. 945–958.

[20] Z. Fan et al. "LSHADE44 with an Improved $\epsilon $ Constraint-Handling Method for Solving Constrained Single-Objective Optimization Problems." 2018 IEEE Congress on Evolutionary Computation (CEC). IEEE, 2018.

[21] R. Polakova, "L-shade with competing strategies applied to constrained optimization," in Evolutionary Computation (CEC), 2017 IEEE Congress on. IEEE, 2017. 1683–1689.

[22] G. Wu, W. Pedrycz, P. N. Suganthan, R. Mallipeddi. A Variable Reduction Strategy for Evolutionary Algorithms Handling Equality Constraints. Applied soft computing, 37 (2015): 774-786.

[23] R. Mallipeddi, P.N. Suganthan, Q.K. Pan, M.F. Tasgetiren, Differential evolution algorithm with ensemble of parameters and mutation strategies, Applied soft computing.11 (2) (2011) 1679–1696.